自动驾驶数据框架（ADDF）安全和操作指南

AWS 规范性指导

Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

AWS 规范性指导: 自动驾驶数据框架（ADDF）安全和操作指南

Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon 的商标和商业外观不得用于任何非 Amazon 的商品或服务，也不得以任何可能引起客户混淆、贬低或诋毁 Amazon 的方式使用。所有非 Amazon 拥有的其他商标均为各自所有者的财产，这些所有者可能附属于 Amazon、与 Amazon 有关联或由 Amazon 赞助，也可能不是如此。

Table of Contents

自动驾驶数据框架（ADDF）安全和操作指南

Amazon Web Services 的 Andreas Falkenberg、Junjie Tang、Torsten Reitemeyer 和 Srinivas Reddy Cheruku

2022 年 11 月（文档历史记录）

自动驾驶数据框架（ADDF）是一个开源项目，旨在为想要实现高级驾驶辅助系统（ADAS）常见任务的汽车团队提供可重复使用的模块化代码构件，例如，配置集中式数据存储、数据处理管道、可视化机制、搜索界面、模拟工作负载、分析界面和预构建的控制面板。通过 ADDF，您可以共享、修改或创建完全可自定义的模块，从而减少创建和部署这些解决方案所需的工作量。

本指南旨在帮助您了解在 AWS Cloud中安全部署和操作 ADDF 的最佳实践。本指南讨论了以下主题：

- 架构和术语 - 查看一般架构、工作流程和重要术语。
- 责任共担模式— 了解您在保护 ADDF 部署和云资源方面的角色和角色。 AWS
- 安全审查流程— 由于 ADDF 是一个开源项目，请查看贡献者如何 AWS 完成安全审查。
- Built-in 安全功能 - 查看如何将安全最佳实践和功能内置到 ADDF 开源项目及其部署框架中。
- 安全设置和操作 - 了解如何在 AWS Cloud中部署和操作 ADDF。

目标受众

本指南适用于负责评估、部署、自定义和操作 ADDF 的开发运营 (DevOps) 团队、基础架构工程师、管理员、IT 安全人员和事件响应团队。您可以将本指南中的建议应用于 proof-of-concept我们的生产环境。

本指南假定您之前不了解 ADDF。但是，我们建议您在继续操作之前先阅读 ADDF 自述文件 (GitHub)。

目标业务成果

本指南旨在帮助您在开发和生产环境中更加自信、安全地设置和操作 ADDF。

ADDF 架构和术语

在理解本指南中的安全和操作主题之前，务必先对自动驾驶数据框架（ADDF）的术语、组件和架构有一个深入的了解。本节包含以下主题：

ADDF 术语

ADDF 的主要术语如下所示：

- ADDF 模块 - 模块是指在高级驾驶辅助系统（ADAS）中实现常见任务的基础设施即代码（IaC）。常见任务包括配置集中式数据存储、数据处理管道、可视化机制、搜索界面、模拟工作负载、分析界面和预构建的控制面板。您可以根据自己的需求创建模块，也可以重复使用或自定义现有模块。

  您可以使用 AWS Cloud Development Kit (AWS CDK) 来定义 ADDF 模块，也可以使用任何常见的 IaC 框架（例如 Hashicorp Terraform 或 AWS CloudFormation）来实现 ADDF 模块。一个模块有一组输入参数。输入参数可能取决于其他模块的输出值。ADDF 模块是 ADDF 目标 AWS 账户最小的部署单元。
- ADDF 部署清单文件 - 该文件定义了独立 ADDF 模块的编排。编排是指模块的部署顺序。在 ADDF 部署清单文件中，您可以使用 ADDF 群组将相关模块分组在一起。在此文件中，您还定义了 ADDF 工具链 AWS 账户、ADDF 目标 AWS 账户和目标。 AWS 区域
- ADDF 部署框架 — 此框架 AWS 账户 根据 ADDF 部署清单文件中定义的编排将 ADDF 模块部署到 ADDF 目标中。ADDF 部署框架是通过使用以下 AWS 开源项目实现的：
  - SeedFarmer(GitHub) — SeedFarmer 是用于 ADDF 部署的 CLI 工具。它管理每个模块的状态，准备和打包模块代码，为 ADDF 部署角色创建最低权限策略，并提供用于部署的语义指 CodeSeeder 令。您可以直接与交互 SeedFarmer 以运行 ADDF 部署，也可以将其集成到持续集成和持续部署 (CI/CD) 管道中。
  - CodeSeeder(GitHub) — 通过 AWS CodeBuild 作业 CodeSeeder 部署任意基础架构作为代码包。SeedFarmer自动编排和运行。 CodeSeeder只能 SeedFarmer 直接与 CodeSeeder之互动。

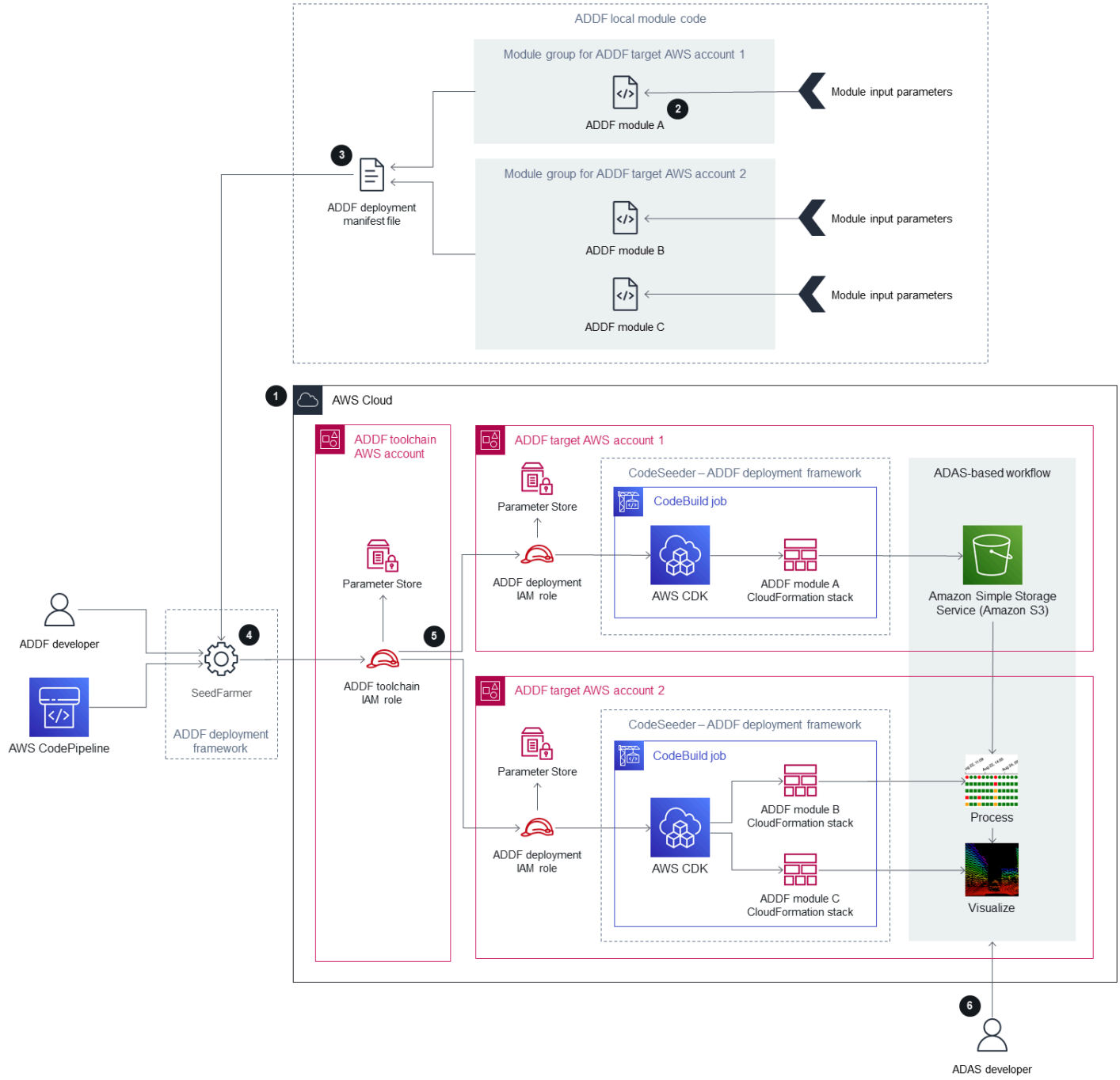
  ADDF 部署框架用来支持单账户和多账户架构中的部署。根据贵组织的需求，您可以决定需要单账户架构还是多账户架构。
- ADDF 工具链 AWS 账户 — 此账户根据 ADDF 部署清单文件中的定义编排和管理将模块部署到 ADDF 目标 AWS 账户中。一个 ADDF 部署只能有一个 ADDF 工具链 AWS 账户。在单账户架构

中，ADDF 工具链 AWS 账户 也是 ADDF 目标 AWS 账户。此账户包含一个名为 ADDF 工具链 IAM 角色的 AWS Identity and Access Management (IAM) 角色，该角色在 ADDF 部署 SeedFarmer 过程中由担任。在本指南中，我们将 ADDF 工具链 AWS 账户 称为工具链账户。

- ADDF 目标 AWS 账户 — 这些是您在其中部署 ADDF 模块的目标帐户。您可以拥有一个或多个目标账户。这些账户包含 ADDF 部署清单文件及其映射模块中描述的资源和应用程序逻辑。在单账户架构中，ADDF 目标也 AWS 账户 是 ADDF 工具链。 AWS 账户每个 ADDF 目标账户都包含一个名为 ADDF 部署 IAM 角色的 IAM 角色，该角色在部署 CodeSeeder 过程中由担任。在本指南中，我们将 ADDF 目标 AWS 账户 称为目标账户。
- ADDF 实例 - 当您在云中部署 ADDF 和模块时（如 ADDF 部署清单文件中所定义），它将成为 ADDF 实例。一个 ADDF 实例可以采用单账户或多账户架构，您可以部署多个 ADDF 实例。有关为您的用例选择实例数量和设计账户架构的更多信息，请参阅 定义您的 ADDF 架构。

ADDF 架构

下图显示了 AWS Cloud中 ADDF 实例的高级架构。它显示了一个多账户架构，包括一个专用工具链账户和两个目标账户。本指南讨论了使用 ADDF 将资源部署到目标账户的端到端流程。



1. 创建并引导 ADDF AWS 账户。

   若要正常运行，必须将每个账户引导至 ADDF 和 AWS CDK。如果这是新的 ADDF 部署，或者您要添加新的目标账户，请执行以下操作：

   a. 工具链账户和每个目标账户 AWS CDK 中的Bootstrap。有关说明，请参阅 Bootstrapping（AWS CDK 文档）。ADDF 使用 AWS CDK 来部署其基础架构。

b. 在工具链账户和每个目标账户中引导 ADDF。有关说明，请参阅《ADDF 部署指南》中的 Bootstrap AWS 账户。这将设置 SeedFarmer 和所需的所有 ADDF-specific IAM 角色 CodeSeeder。

> **Note**
>
> 只有在最初部署 ADDF 或添加新的目标账户时，才需要执行此步骤。该步骤不属于向已建立的 ADDF 实例重复部署 ADDF 的过程。

2. 创建或自定义 ADDF 模块。

   根据您要解决的特定问题创建或自定义 ADDF 模块。您的模块应代表一个独立的任务或一组任务。根据需要定义模块的输入参数，并将模块输出值用作其他模块的输入参数。

3. 在 ADDF 部署清单文件中定义模块编排。

   在 ADDF 清单文件中，将模块整理成组，并定义部署顺序和模块之间的依赖关系。在此文件中，您还可以为每个 ADDF 组及其模块指定单个工具链账户和目标账户（包括 AWS 区域）。

4. 评估 ADDF 部署清单文件并确定部署范围。

   ADDF 开发者或 CI/CD 管道（例如 AWS CodePipeline）通过调用 CLI 工具开始评估 ADDF 部署清单文件。 SeedFarmer若要开始评估：

   - SeedFarmer 使用 ADDF 部署清单文件作为评估的输入参数。
   - 要担任 ADDF 工具链 IAM 角色， SeedFarmer 需要与步骤 1 中在 ADDF 引导过程中定义的相同、有效的 IAM 角色或用户证书。

   如果 SeedFarmer 没有担任 ADDF 工具链 IAM 角色的正确凭证，或者无法访问 ADDF 部署清单文件，则评估将无法开始。

   如果 SeedFarmer 可以开始评估，它将在工具链账户中扮演 ADDF 工具链 IAM 角色。从那里 SeedFarmer 可以访问任何目标账户，只需在该账户中担任 ADDF 部署 IAM 角色即可。SeedFarmer 然后尝试读取工具链账户和目标账户中的任何 ADDF 元数据。将出现以下情况之一：

   - 如果没有 ADDF 元数据可供读取，则表示这是一个新的 ADDF 实例。 SeedFarmer 确定部署范围是整个 ADDF 部署清单文件及其内容。
   - 如果存在 ADDF 元数据，则将 ADDF 部署清单文件及其内容与目标账户中现有已部署项目的 MD5 哈希值进行 SeedFarmer 比较。如果检测到可部署的更改，该过程将继续。如果未检测到可部署的更改，该过程将结束。

5. 将范围内 ADDF 模块部署到目标账户。

CodeSeeder 根据 ADDF 部署清单文件和上一步的评估结果，现在有待运行的部署的有序列表。根据该排序列表，在每个关联 CodeSeeder 的目标账户中担任 ADDF 部署 IAM 角色。然后，它 CodeSeeder 在 AWS CodeBuild 作业中运行，为 ADDF 模块创建或更新各个 IaC 部署（例如 AWS CDK 应用程序）。默认情况下，ADDF 使用 AWS CDK 作为其 IaC 框架，但也支持其他常见的 IaC 框架。完成每个目标账户的流程后，您将拥有一个完全部署、跨账户、端到端 ADAS-based 的工作流程，如您在 ADDF 部署清单文件中所定义。

如果您使用单账户架构，则工具链账户和目标账户是同一个账户，并且一个账户具有上述所有功能。

6. 使用 ADDF-deployed基础架构。

   ADAS 开发人员可以使用由您的用例定义的已部署 ADAS-based 工作流程。

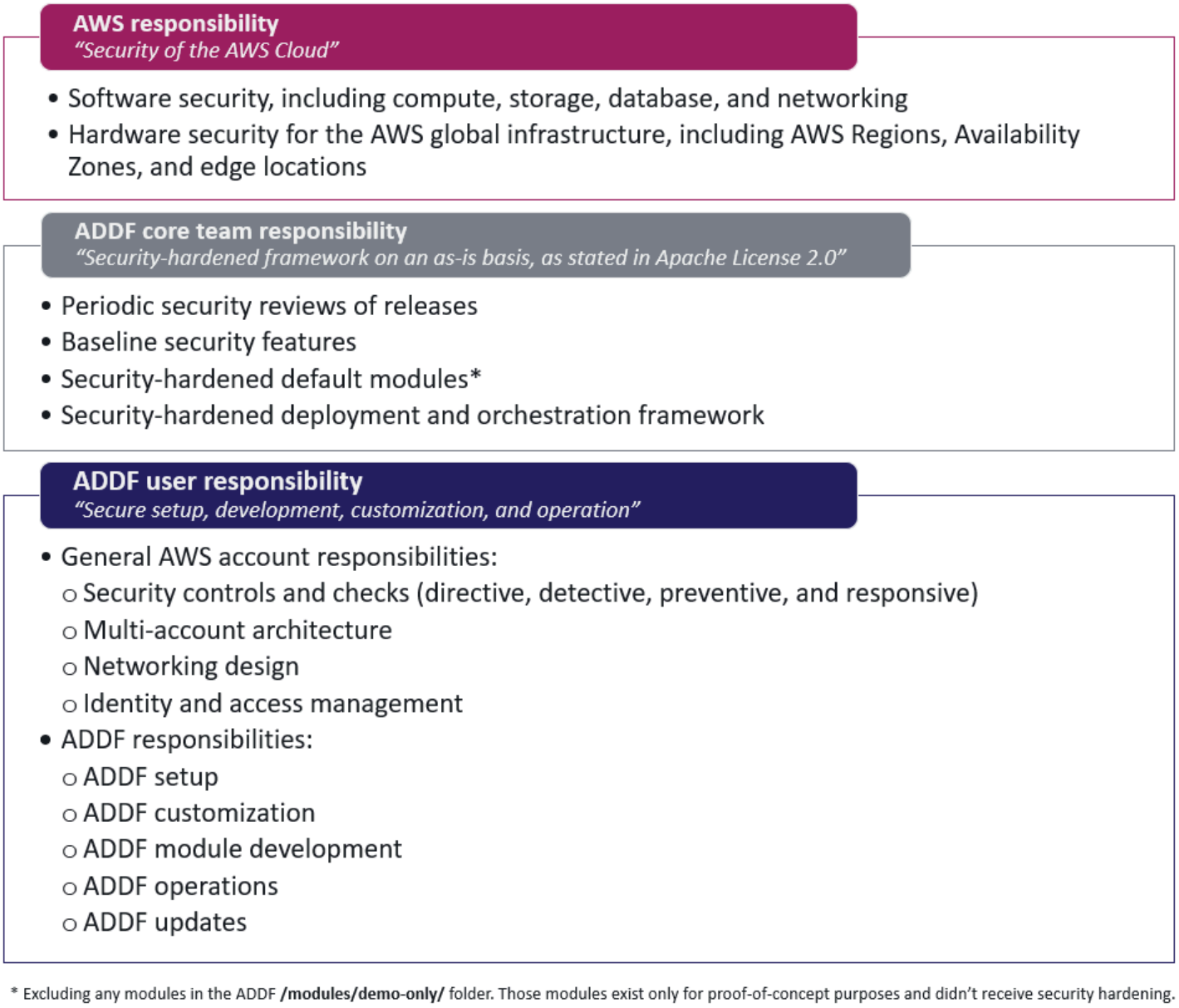
   该工作流描述了 ADDF 多账户环境的单个实例的架构。根据您的开发、部署和操作模型，我们建议您在多阶段环境中运行多个 ADDF 实例。典型的设置可能包括一个专 AWS 账户 用于每个部署阶段的 ADDF 实例，例如用于开发、测试和生产的分支。假设您为每个 ADDF 实例创建了唯一的资源命名空间，您也可以在同 AWS 区域一个单账户或多账户环境中运行多个 ADDF 实例。有关更多信息，请参阅 定义您的 ADDF 架构。

ADDF 责任共担模式

适用的责任共担模型 AWS 服务 也适用于自动驾驶数据框架 (ADDF)。如下图所示，下列实体共同承担保护 ADDF 的责任：

- AWS— 云基础架构提供商产品 AWS 服务。
- ADDF 核心团队 — ADDF 核心团队是在 ADDF 存储库中发布 ADDF 版本的实体 ()。GitHub
- ADDF 用户 - ADDF 用户包括但不限于：
  - ADDF 开发人员 - 任何更改、自定义或创建新 ADDF 模块代码的人。
  - ADDF 操作员 - 任何设置和操作 ADDF 实例的人。
  - ADAS 开发人员 - ADDF 部署的资源的最终用户或使用者。例如，ADAS 开发人员可以查询作为 ADDF 部署的一部分创建的可视化前端。

下图总结了 ADDF 核心团队和 ADDF 用户之间的 AWS共同责任。



* Excluding any modules in the ADDF **/modules/demo-only/** folder. Those modules exist only for proof-of-concept purposes and didn't receive security hardening.

AWS 责任

AWS 负责保护运行中提供的所有服务的基础架构 AWS Cloud，如AWS 分担责任模式所定义。该基础架构由运行 AWS Cloud 服务的硬件、软件、网络和设施组成。

ADDF 核心团队责任

根据 A pache License 2.0 ()，ADDF 核心团队在尽最大努力的基础上提供了一个本身安全的框架。GitHubADDF 核心团队负责以下工作：

- 定期对版本进行安全审查
- 基线安全功能
- Security-hardened 默认模块（这不包括/modules/demo-only/文件夹中的任何模块。 这些模块仅用于概念验证目的，不接受安全强化。 ）
- Security-hardened 部署和编排框架

这些安全责任仅适用于 GitHub 存储库中提供的框架，无需进行任何修改或自定义。这包括所有 ADDF 模块，modules/demo-only/ 文件夹中的 ADDF 模块除外。该文件夹中的 ADDF 模块未经过安全加固，不应部署在生产环境或任何包含敏感或受保护数据的环境中。包含这些模块是为了展示系统功能，您可以将它们作为创建自己的自定义安全加固模块的基础。

> **Note**
>
> ADDF 作为一个框架按原样交付。正如 Apache 许可证 2.0 (GitHub) 中所述，它不附带任何责任和担保。您应自行对 ADDF 进行安全评测，验证其是否符合贵组织的特定安全要求。

ADDF 用户责任

只有以安全的方式设置、自定义和操作 ADDF 时，ADDF 及其模块才是安全的。ADDF 用户对以下方面的安全负全部责任：

- 一般 AWS 账户 职责：
  - 安全控制和检查（指令性、侦测性、预防性和响应性）
  - Multi-account 建筑
  - 网络设计
  - Identity and access management
- ADDF-specific 职责：
  - ADDF 设置
  - ADDF 自定义
  - ADDF 模块开发
  - ADDF 操作
  - ADDF 更新

General AWS 账户 责任

在将任何 ADDF-related 资源部署到中之前 AWS 账户， AWS 账户 应根据AWS Well-Architected 框架中的最佳实践进行配置。这包括指导性、检测性、预防性和响应性安全控制。您应该制定详细的缓解流程，以防出现任何安全违规或事件。贵组织的策略应涵盖集中管理身份、访问权限和网络的要求。通常，这些要求和服务由专门的登录区团队处理。

ADDF-specific 责任

安全设置 ADDF

ADDF 用户的责任始于根据 ADDF 文档对 ADDF 进行安全设置。我们强烈建议您按照 ADDF 部署指南（GitHub）中的说明进行操作。有关安全设置 ADDF 的更多信息，请参阅 定义您的 ADDF 架构 和 初始 设置。

安全自定义 ADDF

如果对 ADDF 核心功能（例如 CodeSeeder SeedFarmer、和 ADDF 核心模块）进行自定义，则 ADDF 用户对这些更改承担全部责任。有关更多信息，请参阅 自定义 ADDF 部署框架代码。

安全开发 ADDF 模块

ADDF 用户对使用 ADDF 部署的任何自定义模块负全部责任。此外，ADDF 用户应对 ADDF-supplied 模块的任何代码更改负责。有关更多信息，请参阅 在 ADDF 中编写自定义模块。

安全更新和操作 ADDF

随着框架的演变，ADDF 会收到功能和安全更新。ADDF 用户有责任定期检查发布到 GitHub 存储库的更新，并长期安全地运行 ADDF。有关更多信息，请参阅 重复部署 ADDF、重复进行安全审计、ADDF 更新 和 停用。

ADDF 安全审查流程

自动驾驶数据框架（ADDF）在构建时充分考虑了安全性。在向公众发布之前， AWS 对 ADDF 进行了初步的内部安全审查，解决了所有已发现的安全问题。两者 AWS 以及开源社区都为该框架的持续安全审查做出了贡献。

定期进行安全审查 AWS

ADDF 由所有的 awslabs GitHub 组织发布。 AWS AWS 定期对该组织中的代码执行自动和手动安全审查，以尽最大努力验证安全性。根据 AWS 政策， AWS 不披露有关安全审查频率、方法或所用工具的信息。此外， AWS 不发布任何有关ADDF的内部审计报告。不过，任何已发现的安全调查结果都将通过拉取请求紧急修复和发布。

> **Note**
>
> ADDF 作为框架在 “按现状” 的基础上交付，不附带任何明示或暗示的担保或条件，包括但不限于 Apache License 2.0 () 中所述的所有权、非侵权、适销性或适用于特定目的的任何担保或条件。 GitHub您应自行对 ADDF 进行安全评测，验证其是否符合贵组织的特定安全要求，正如 Apache 许可证 2.0 中规定的那样，您自行负责确定使用或重新分发 ADDF 的适当性，并承担您在该许可证下的行为或权限相关的任何风险。

开源安全审查和贡献

ADDF 是一个开源项目，每个人都可以做出贡献。我们邀请所有用户对自己的框架进行安全审查，并报告调查结果的任何安全问题。如果您在代码中发现问题，请遵循安全问题通知（ADDF 文档）中的指南。

ADDF 内置安全功能

自动驾驶数据框架（ADDF）具有各种内置安全功能。默认情况下，这些功能旨在帮助您建立安全框架，并帮助贵组织满足常见的企业安全要求。

以下是内置安全功能：

- ADDF 模块代码的最低权限
- 基础设施即代码
- IaC 自动安全检查
- 的自定义最低权限策略 AWS CDK 部署角色
- Least-privilege 模块部署规范文件的策略
- 数据加密
- 使用 Secrets Manager 存储凭证
- SeedFarmer 和的安全审查 CodeSeeder
- 对的权限边界支持 AWS CodeBuild 的角色 CodeSeeder
- AWS 多账户架构
- Least-privilege 多账户部署权限

ADDF 模块代码的最低权限

最低权限是授予执行任务所需的最低权限的安全最佳实践。有关更多信息，请参阅应用最低权限权限。ADDF-provided 模块在其代码和部署的资源中严格遵循最低权限原则，如下所示：

- 为 ADDF 模块生成的所有 AWS Identity and Access Management (IAM) 策略都具有该用例所需的最低权限。
- AWS 服务 是根据最低权限原则配置和部署的。 ADDF-provided 模块仅使用特定用例所需的服务和服务功能。

基础设施即代码

ADDF 作为一个框架，用于将 ADDF 模块部署为基础设施即代码（IaC）。IaC 消除了手动部署过程，有助于防止手动过程可能导致的错误和配置错误。

ADDF 旨在使用任何常见的 IaC 框架来编排和部署模块。这包括但不限于：

- AWS Cloud Development Kit (AWS CDK)
- AWS CloudFormation
- Hashicorp Terraform

您可以使用不同的 IaC 框架来编写不同的模块，然后使用 ADDF 来部署它们。

ADDF 模块使用的默认 IaC 框架是。 AWS CDK AWS CDK 是一个面向对象的高级抽象，你可以用它来强制性地定义 AWS 资源。 AWS CDK 默认情况下，已经针对各种服务和场景强制实施了安全最佳实践。通过使用 AWS CDK，可以降低安全配置错误的风险。

IaC 自动安全检查

开源 cdk-nag 实用程序 (GitHub) 已集成到 ADDF 中。此实用程序会自动检查基于 AWS CDK 的 ADDF 模块是否符合一般和安全最佳实践。cdk-nag 实用工具使用规则和规则包来检测和报告违反最佳实践的代码。有关规则和完整列表的更多信息，请参阅 cdk-nag 规则 ()。GitHub

的自定义最低权限策略 AWS CDK 部署角色

ADDF 广泛使用了 AWS CDK v2。需要将所有 ADDF AWS 账户 引导到。 AWS CDK有关更多信息，请参阅 Bootstrapping（AWS CDK 文档）。

默认情况下，将许可AdministratorAccess AWS 托管策略 AWS CDK 分配给在引导账户中创建的 AWS CDK 部署角色。此角色的全名是`cdk-[CDK_QUALIFIER]-cfn-exec-role-[AWS_ACCOUNT_ID]-[REGION]`。 AWS CDK AWS 账户 作为部署过程的一部分，使用此角色将资源部署到引导程序中 AWS CDK 。

根据贵组织的安全需求，`AdministratorAccess` 策略可能过于宽松。作为 AWS CDK 引导过程的一部分，您可以根据需要自定义策略和权限。您可以使用 `--cloudformation-execution-policies` 参数通过新定义的策略重新引导账户，来更改策略。有关更多信息，请参阅自定义引导（AWS CDK 文档）。

> **Note**
>
> 此安全功能并非 ADDF 所特有，但在本节中列出它是因为其可以提高 ADDF 部署的整体安全性。

Least-privilege 模块部署规范文件的策略

每个模块都包含一个名为 deployspec.yaml 的部署规范文件。此文件定义了模块的部署说明。CodeSeeder 使用它在目标账户中部署已定义的模块 AWS CodeBuild。 CodeSeeder 按照部署规范文件中的说明， CodeBuild 为分配默认服务角色来部署资源。该服务角色是根据最低权限原则设计的。它包括部署 AWS CDK 应用程序所需的所有权限，因为所有 ADDF-provided 模块都是作为 AWS CDK 应用程序创建的。

但是，如果您需要在之外运行任何阶段命令 AWS CDK，则需要创建自定义 IAM 策略，而不是使用默认的服务角色 CodeBuild。例如，如果您使用的是 IaC 部署框架以外的 IaC 部署框架 AWS CDK，例如 Terraform，则需要创建一个 IAM 策略，为该特定框架提供足够的权限才能正常运行。另一种需要专用 IAM 策略的场景是，在`install`、`pre_buildbuild`、或`post_build`暂存命令 AWS 服务 中包含对其他人的直接 AWS Command Line Interface (AWS CLI) 调用。例如，如果您的模块包含用于将文件上传到 S3 存储桶的 Amazon Simple Storage Service（Amazon S3）命令，则需要自定义策略。自定义 IAM 策略为部署之外的任何 AWS 命令提供精细控制。 AWS CDK 为 ADDF 模块创建自定义 IAM policy 时，确保应用最低权限许可。

数据加密

ADDF 会存储和处理潜在的敏感数据。为了帮助保护这些数据， SeedFarmer CodeSeeder、和 ADDF-provided 模块会对所有使用的静态和传输中的数据进行加密 AWS 服务 （除非对`demo-only`文件夹中的模块另有明确说明）。

使用 Secrets Manager 存储凭证

ADDF 处理不同服务的各种机密，例如 Docker Hub 和 Amazon Red JupyterHub sh ift。ADDF AWS Secrets Manager用于存储任何 ADDF-related 机密。这可以帮助您从源代码中删除敏感数据。

Secrets Manager 密钥仅存储在目标账户中，以满足该账户正常运行的需要。默认情况下，工具链账户不包含任何密钥。

SeedFarmer 和的安全审查 CodeSeeder

SeedFarmer和 CodeSeeder（GitHub存储库）用于部署 ADDF 及其 ADDF 模块。这些开源项目要经过与 ADDF 相同的定期 AWS 内部安全审查流程，如中所述。ADDF 安全审查流程

对的权限边界支持 AWS CodeBuild 的角色 CodeSeeder

IAM 权限边界是一种常见的安全机制，它定义了基于身份的策略可以向 IAM 实体授予的最大权限。SeedFarmer 并 CodeSeeder 支持每个目标账户的 IAM 权限边界附件。权限边界限制了 CodeSeeder 部署模块 CodeBuild 时使用的任何服务角色的最大权限。IAM 权限边界必须由安全团队在 ADDF 外部创建。IAM 权限边界策略附件作为 ADDF 部署清单文件 deployment.yaml 中的一个属性被接受。有关更多信息，请参阅权限边界（SeedFarmer 文档）。

工作流程如下所示：

1. 您的安全团队根据您的安全需求定义和创建 IAM 权限边界。必须在每个 ADDF AWS 账户中单独创建 IAM 权限边界。输出是权限边界策略 Amazon 资源名称（ARN）列表。
2. 安全团队与您的 ADDF 开发团队共享策略 ARN 列表。
3. ADDF 开发人员团队将策略 ARN 列表集成到清单文件中。有关此集成的示例，请参阅 sam ple-permissionboundary.yaml ()。 GitHub
4. 成功部署后，权限边界将附加到 CodeBuild 用于部署模块的所有服务角色。
5. 安全团队会根据需要监控是否应用权限边界。

AWS 多账户架构

正如 AWS Well-Architected 框架的安全支柱所定义的那样，根据组织的要求将资源和工作负载分成多个 AWS 账户资源和工作负载被认为是最佳实践。这是因为 AWS 账户 充当隔离边界。有关更多信息，请参阅 AWS 账户 management and separation。这一概念的实现称为多账户架构。与单账户架构相比，设计合理的 AWS 多账户架构可提供工作负载分类，在发生安全漏洞时减小影响范围。

ADDF 原生支持 AWS 多账户架构。您可以根据需要将 ADDF 模块分发给多个 AWS 账户 以满足组织安全和职责分工要求的模块。您可以将 ADDF 部署到单个 AWS 账户，并结合工具链和目标账户功能。或者，您可以为 ADDF 模块或模块组创建单独的目标账户。

您需要考虑的唯一限制是，ADDF 模块代表每个 AWS 账户模块的最小部署单元。

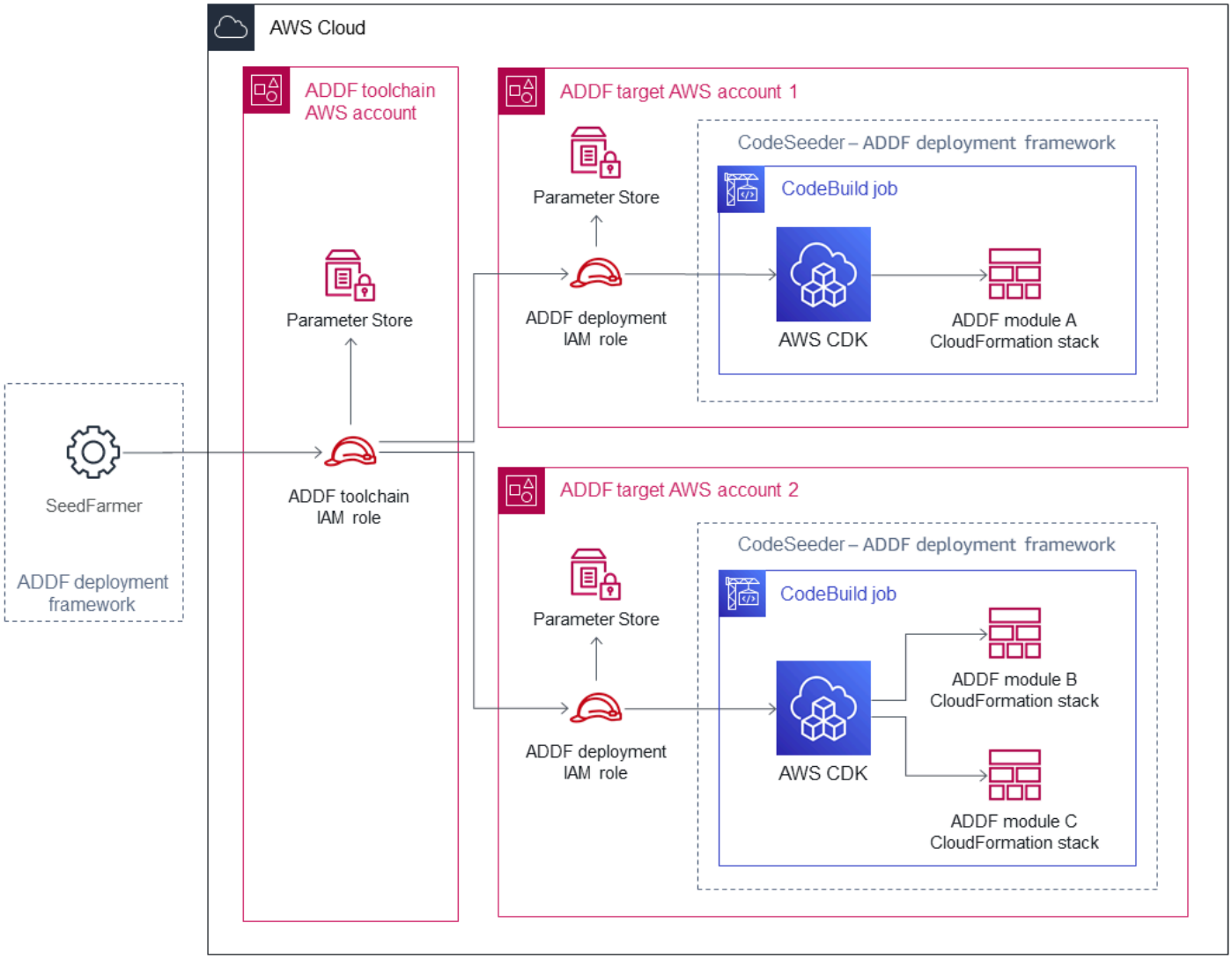
对于生产环境，建议您使用由一个工具链账户和至少一个目标账户组成的多账户架构。有关更多信息，请参阅 ADDF 架构。

Least-privilege 多账户部署权限

如果您使用多账户架构，则 SeedFarmer 需要访问目标账户才能执行以下三个操作：

1. 将 ADDF 模块元数据写入工具链账户和目标账户。
2. 从工具链账户和目标账户读取当前的 ADDF 模块元数据。
3. 在目标账户中启动 AWS CodeBuild 任务，以部署或更新模块。

下图显示了跨账户关系，包括代任 ADDF-specific AWS Identity and Access Management (IAM) 角色的操作。



这些跨账户操作是使用明确定义的 assume-role 操作实现的。

- ADDF 工具链 IAM 角色部署在工具链账户中。 SeedFarmer 担任这个角色。该角色具有执行 `iam:AssumeRole` 操作的权限，并且可以在每个目标账户中承担 ADDF 部署 IAM 角色。此外，ADDF 工具链 IAM 角色可以运行本地 AWS Systems Manager 参数存储操作。

- ADDF 部署 IAM 角色部署在每个目标账户中。只能使用 ADDF 工具链 IAM 角色从工具链账户承担此角色。此角色有权运行本地 P AWS Systems Manager arameter Store AWS CodeBuild 操作，并有权运行启动和描述 CodeBuild 任务的操作 CodeSeeder。

这些 ADDF-specific IAM 角色是作为 ADDF-bootstrapping流程的一部分创建的。有关更多信息，请参阅《ADDF 部署指南》 AWS 账户 (GitHub) 中的 Bootstrap。

所有跨账户权限都是根据最低权限原则设置的。如果一个目标账户遭到入侵，对另一个 ADDF AWS 账户的影响很小或没有影响。

对于 ADDF 的单账户架构，角色关系保持不变。只是折叠成一个 AWS 账户。

ADDF 安全设置和操作

自动驾驶数据框架 (ADDF) 应被视为一款自定义软件，需要组织中的专门安全团队进行持续维护 DevOps 和保养。本节介绍了与安全相关的常见任务，这些任务可帮助您在 ADDF 的整个生命周期中设置和操作 ADDF。

本节包含以下任务：

- 定义您的 ADDF 架构
- 初始 设置
- 自定义 ADDF 部署框架代码
- 在 ADDF 中编写自定义模块
- 重复部署 ADDF
- 重复进行安全审计
- ADDF 更新
- 停用

定义您的 ADDF 架构

ADDF 实例的安全性取决于它所部署的 AWS 账户 环境。此 AWS 账户 环境的设计必须满足您的特定用例的安全和操作需求。例如，在概念验证（PoC）环境中设置 ADDF 实例的安全和操作相关任务和注意事项与在生产环境中设置 ADDF 的任务和注意事项不同。

在 PoC 环境中运行 ADDF

如果您打算在 PoC 环境中使用 ADDF，我们建议您创建一个不包含任何其他 AWS 账户 工作负载的 ADDF 专用。这有助于在您探索 ADDF 及其功能时确保您的账户安全。这种方法的优点如下：

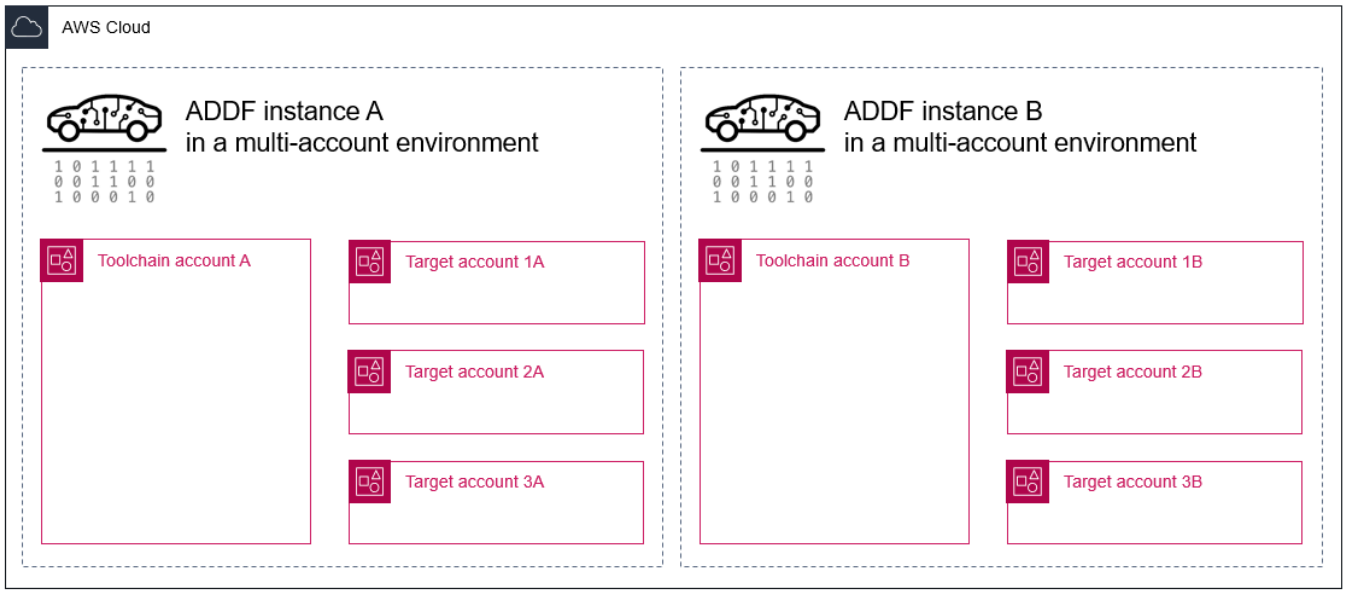
- 如果出现严重的 ADDF 配置错误，不会对其他工作负载产生不利影响。
- 不存在任何其他可能对 ADDF 设置产生不利影响的工作负载配置错误的风险。

即使对于 PoC 环境，我们仍然建议您尽可能遵循 在生产环境中运行 ADDF 中列出的最佳实践。

在生产环境中运行 ADDF

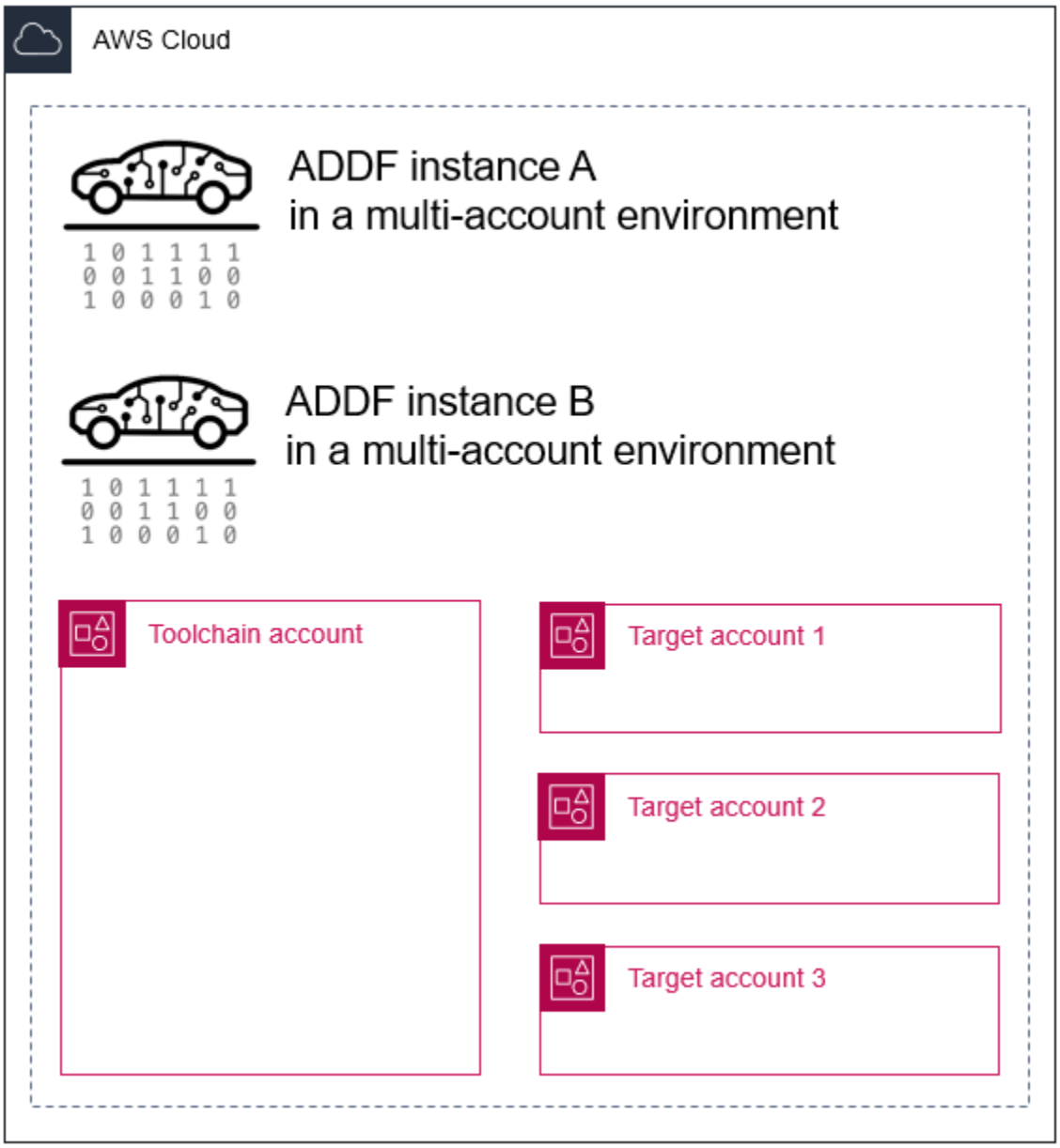
如果您要在企业生产环境中使用 ADDF，我们强烈建议您考虑组织的安全最佳实践，并相应地实施 ADDF。除了组织的安全最佳实践外，我们还建议您实施以下措施：

- 创建一支长期的、敬业的 ADDF DevOps 团队 ——ADDF 需要被视为一款定制软件。它需要专门的 DevOps团队进行持续的维护和保养。在开始在生产环境中运行 ADDF 之前，应定义一个具有足够规模和能力的 DevOps团队，并投入全部资源，直到 ADDF 部署的生命周期结束为止。
- 使用多账户架构 - 每个 ADDF 实例都应部署在自己专用的 AWS 多账户环境中，没有任何其他不相关的工作负载。根据AWS 客户管理和分离（Fr AWS Well-Architected amework）中的定义，根据组织的要求将资源和工作负载分成多个 AWS 账户资源和工作负载被认为是最佳实践。这是因为 AWS 账户 充当隔离边界。与单账户架构相比，设计合理的 AWS 多账户架构可提供工作负载分类，在发生安全漏洞时减小影响范围。使用多账户架构还有助于您的账户保持在 AWS 服务 限额内。根据组织的安全和职责分离要求，将您的 ADDF 模块分发到多个 AWS 账户 中。
- 部署多个 ADDF 实例 - 根据需要设置任意数量的单独 ADDF 实例，以便根据组织的软件开发流程正确开发、测试和部署 ADDF 模块。在设置多个 ADDF 实例时，可使用以下方法之一：
  - 不同多 AWS 账户环境中的多个 ADDF 实例 — 您可以单独使用 AWS 账户 来隔离不同的 ADDF 实例。例如，如果您的组织有专门的开发、测试和生产阶段，则可以为每个阶段创建单独的 ADDF 实例和专用账户。这样做有很多好处，比如降低错误跨阶段传播的风险，帮助您实施审批流程，以及限制用户只能访问某些环境。下图显示了在单独的多账户环境中部署的两个 ADDF 实例。



  - 同一多账户环境中的 AWS 多个 ADDF 实例-您可以创建共享相同多账户环境的多个 ADDF 实例。AWS 这实际上是在同一 AWS 账户中创建独立的分支。例如，如果不同的开发人员并行工作，则开发人员可以在同一 AWS 账户中创建专用的 ADDF 实例。这有助于开发人员在独立的分支中

进行开发和测试。如果使用这种方法，对于每个 ADDF 实例，ADDF 资源必须具有唯一的资源名称。默认情况下，ADDF 预提供的模块支持此功能。只要不超过 AWS 服务 限额，就可以使用这种方法。下图显示了在共享的多账户环境中部署的两个 ADDF 实例。



- 同一 AWS 单账户环境中的多个 ADDF 实例 – 此架构与前面的示例非常相似。不同之处在于，多个 ADDF 实例部署在单账户环境中，而不是多账户环境中。此架构适合非常简单的 ADDF 用例，这些用例的范围非常有限，而且多个开发人员同时在不同的分支上工作。

由于 SeedFarmer 是控制 ADDF 实例部署的单一工具，因此您可以构建适合组织部署策略和 CI/CD 流程的任何环境和账户架构。

- 根据组织的安全要求自定义 AWS Cloud Development Kit (AWS CDK) 引导流程-默认情况下，在引导过程中 AWS CDK 分配AdministratorAccess AWS 托管策略。此策略授予完全管理权限。如果此策略对于贵组织的安全要求而言过于宽松，您可以自定义应用哪些策略。有关更多信息，请参阅 的自定义最低权限策略 AWS CDK 部署角色。
- 在 IAM 中设置访问权限时遵循最佳实践 — 建立允许用户访问 ADDF AWS 账户的结构化 AWS Identity and Access Management (IAM) 访问解决方案。ADDF 框架的设计遵循最低权限原则。您的 IAM 访问模式还应遵循最低权限原则，应符合贵组织的要求，并遵守 IAM 安全最佳实践（IAM 文档）。
- 根据组织的最佳实践设置网络 - ADDF 包含一个可选的网络 AWS CloudFormation 堆栈，用于创建基本的公有或私有虚拟私有云（VPC）。根据组织的配置，此 VPC 可能会将资源直接公开到互联网。我们建议您遵循组织的最佳网络实践，创建一个自定义的安全加固网络模块。
- 在 AWS 账户 级别部署安全预防、检测和缓解措施 —— AWS 提供各种安全服务，例如亚马逊、GuardDuty AWS Security Hub CSPM、Amazon Detective 和 AWS Config。在 ADDF 中启用这些服

务，AWS 账户 并集成组织的安全预防、检测、缓解和事件处理流程。我们建议您遵循安全、身份和合规性最佳实践（AWS 架构中心）以及该服务文档中包含的任何特定于服务的建议。有关更多信息，请参阅 AWS Security Documentation。

ADDF 未涉及这些主题，因为实施和配置细节在很大程度上取决于贵组织的具体要求和流程。这些主题主要由贵组织负责阐述。通常，管理 AWS 登录区的团队会帮助您规划和实施 ADDF 环境。

初始 设置

根据 ADDF 部署指南 () GitHub 设置 ADDF。任何部署的起点都是自动驾驶数据框架 Git Hub 存储库中的 `/manifest` 文件夹。`/manifest/example-dev` 文件夹包含一个用于演示的示例部署。以本示例为起点，设计您自己的部署。在该目录中，有一个名为 deployment.yaml 的 ADDF 部署清单文件。它包含用于 SeedFarmer 管理、部署或删除 ADDF 及其资源的所有信息。 AWS Cloud您可以在专用文件中创建 ADDF 模块组。core-modules.yaml 是核心模块组的一个示例，它包含 ADDF 提供的所有核心模块。总之，deployment.yaml 文件包含对将部署到其目标账户的组和模块的所有引用，并指定部署顺序。

为了实现安全和合规的配置，尤其是在非概念验证环境中，我们建议您查看要部署的每个模块的源代码。根据安全加固最佳实践，您应该只部署预期用例所需的模块。

> **Note**
>
> `modules/demo-only/` 文件夹中的 ADDF 模块未经过安全加固，不应部署在生产环境或任何包含敏感或受保护数据的环境中。包含这些模块是为了展示系统功能，您可以将它们作为创建自己的自定义安全加固模块的基础。

自定义 ADDF 部署框架代码

ADDF 部署框架及其编排和部署逻辑可以完全自定义，以满足任何要求。但是，出于以下原因，我们建议您不要自定义或尽量减少更改：

- 保持上游兼容性 - 上游兼容性使更新 ADDF 以获取最新功能和安全更新变得更容易。更改框架会破坏与 SeedFarmer CodeSeeder、和任何 ADDF 核心模块的原生向后兼容性。
- 安全后果 - 更改 ADDF 部署框架可能是一项复杂的任务，会带来意想不到的安全后果。在最坏的情况下，框架更改可能会产生安全漏洞。

在可能的情况下，构建和自定义自己的模块代码，而不是修改 ADDF 部署框架和 ADDF 核心模块代码。

> **Note**
>
> 如果您认为 ADDF 部署框架的特定部分需要改进或进一步加强安全性，请通过拉取请求将您的更改提交到 ADDF 存储库。有关更多信息，请参阅 开源安全审查和贡献。

在 ADDF 中编写自定义模块

创建新的 ADDF 模块或扩展现有模块是 ADDF 的核心概念。在创建或自定义模块时，我们建议您遵循一般 AWS 安全最佳实践和组织的安全编码最佳实践。此外，我们建议您根据组织的安全要求进行初步和定期的内部或外部技术安全审查，以进一步降低出现安全问题的风险。

重复部署 ADDF

按照 ADDF 部署指南 () GitHub 中所述部署 ADDF 及其模块。为了支持在目标账户中添加、更新或删除资源的重复进行 ADDF 部署，请 SeedFarmer 使用存储在工具链和目标账户的 Parameter Store 中的 MD5 哈希值将当前部署的基础架构与本地代码库清单文件中定义的基础架构进行比较。

这种方法遵循的 GitOps 范式是，您的源存储库（您操作的本地代码库 SeedFarmer）是真实来源，而在其中明确声明的基础架构是部署的预期结果。有关的更多信息 GitOps，请参阅什么是 GitOps（GitLab 网站）。

重复进行安全审计

像组织中的任何其他软件一样，将 ADDF 和自定义 ADDF 模块代码集成到安全风险管理、安全审查和安全审计周期中。

ADDF 更新

作为其持续开发工作的一部分，ADDF会定期收到更新。这包括功能更新以及与安全相关的改进和修复。我们建议您定期检查新的框架版本，并及时应用更新。有关更多信息，请参阅 Steps to update ADDF（ADDF 文档）。

停用

如果不再需要 ADDF，从您的 AWS 账户中删除 ADDF 及其所有相关资源。任何无人值守和未使用的基础设施都会产生不必要的成本，带来潜在的安全风险。有关更多信息，请参阅 Steps to destroy ADDF（ADDF 文档）。

后续步骤

本指南回顾了在您的 AWS Cloud 环境中部署自动驾驶数据框架 (ADDF) 时的安全和运营最佳实践和注意事项。本指南回顾了 ADDF 用户和 ADDF 核心团队之间的分担责任模型， AWS 以便您了解自己在安全设置和操作 ADDF 方面的角色和责任。它还包括在 ADDF 生命周期中安全操作 ADDF 的建议，包括针对特定环境的建议。

我们建议您熟悉 资源 一节中的资源。准备就绪后，可以按照 ADDF 部署指南 () GitHub 中的说明设置 ADDF。

在设置和操作 ADDF 时，如果您认为部署框架需要改进或进一步加强安全性，请通过拉取请求将您的更改提交到 ADDF 存储库。有关更多信息，请参阅 开源安全审查和贡献。

资源

AWS 文档

- 使用 ADDF 开发和部署自定义工作流程 AWS（AWS 博客文章）
- AWS 安全服务文档
- IAM 安全最佳实操
- AWS 账户管理和分离
- 引导用于 AWS CDK
- AWS 责任共担模式
- AWS 架构完善的框架

开源资源

- ADDF 存储库 () GitHub
- ADDF 部署指南 () GitHub
- CodeSeeder存储库 (GitHub)
- SeedFarmer存储库 (GitHub)

注意事项

客户有责任对本文档中的信息进行单独评测。本文件：(a) 仅供参考，(b) 代表当前 AWS 的产品供应和做法，如有更改，恕不另行通知，以及 (c) 不产生其关联公司、供应商或许可方的任何承诺或保证。AWS AWS 产品或服务 “按原样” 提供，不附带任何形式的担保、陈述或条件，无论是明示还是暗示。

对客户的责任和责任由 AWS 协议控制，本文档不属于其客户之间的任何协议，也不会对其 AWS 进行修改。 AWS

© 2022，Amazon Web Services, Inc. 或其附属公司。保留所有权利。

文档历史记录

下表介绍了本指南的一些重要更改。如果您希望收到有关未来更新的通知，可以订阅 RSS 源。

变更	说明	日期
初次发布	—	2022 年 11 月 15 日

本文属于机器翻译版本。若本译文内容与英语原文存在差异，则一律以英文原文为准。